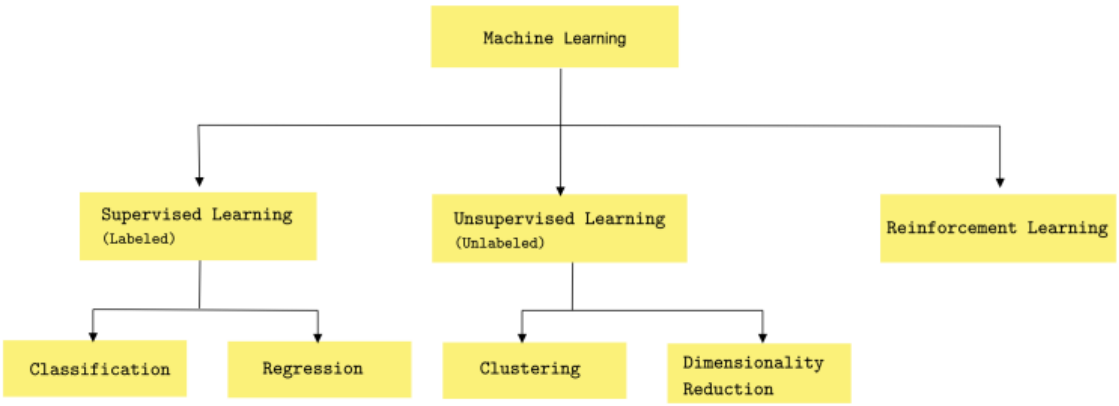
Model Development and Evaluation

November 27, 2024

1 Types of machine learning

Machine learning (ML) is a field of computer science that deals with the creation of algorithms that can discover patterns by themselves without being explicitly programmed. There are different types of ML algorithms, and these are categorized into three different categories, as shown in the following diagram:



As shown in the preceding diagram, there are three different categories of ML algorithms:

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

1.1 Supervised learning

The primary objective of supervised learning is to generalize a model from labeled training data. Once a model has been trained, it allows users to make predictions about unseen future data. Here, by labeled training data, we mean the training examples know the associated output labels. Hence, it is referred to as supervised learning. The learning process can be thought of as a teacher supervising the entire process. In such a learning process, we know the correct answer initially, and the students learn enough iteratively over time and try to answer unseen questions. The errors in the answers are corrected by the teacher. The process of learning stops when we can ensure the performance of the student has reached an acceptable level.

In supervised learning, we have input variables (xi) and output variables (Yi). With this, we can learn a function, f, as shown by the following equation:

$$Y_i = f(x_i)$$

The objective is to learn a general mapping function, f, so that the function can predict the output variable, Y, for any new input data, x. Supervised learning algorithms can be categorized into two groups, as follows:

1. Regression
2. Classification

1.2 Regression

A regression problem has an output variable or dependent variable. This is a real value, such as weight, age, or any other real numbers.

1.3 Classification

A classification problem has the output variable in the form of a category value; for example, red or white wines; young, adult, or old. For classification problems, there are different types of classification algorithms. Some of the most popular ones are as follows: Linear classifier: Naive Bayes classifier, logistic regression, linear SVM

1. Nearest neighbor
2. Decision tree classifier
3. Decision tree classifier
4. Support vector machines
5. Random Forest classifier
6. Neural network classifiers
7. Boosted trees classifier

1.4 Unsupervised learning

Unsupervised machine learning deals with unlabeled data. This type of learning can discover all kinds of unknown patterns in the data and can facilitate useful categorization. Consider a scenario where patients use an online web application to learn about a disease, learn about their symptoms, and manage their illness. Such web applications that provide psychoeducation about certain diseases are referred to as Internet-Delivered Treatments (IDT). Imagine several thousand patients accessing the website at different timestamps of the day, learning about their illness, and all their activities are being logged into our database. When we analyze these log files and plot them using a scatter plot, we find a large group of patients who are accessing the website in the afternoon and a large chunk accessing the website in the evening. Some other patients also follow random login patterns. This scenario illustrates two distinct clusters of patients: one active in the afternoon and one active in the evening. This typical scenario is an example of a clustering task. There are several types of unsupervised learning algorithms that we can use. However, two major unsupervised learning tasks are clustering and dimensionality reductions.

1.4.1 Applications of unsupervised learning

There are several applications of unsupervised learning algorithms. Let's take a look at a few here:

1. Clustering: These types of algorithms allow us to categorize the dataset into several similar groups, referred to as a cluster. Each cluster represents a group of similar points.
2. Association mining: These types of unsupervised learning algorithms allow us to find frequently occurring items in our dataset.
3. Anomaly detection: These types of unsupervised learning algorithms help us to determine unusual data points in any existing dataset.
4. Dimensionality reduction: These techniques are commonly used in data processing in order to reduce the number of features in a dataset. This is one of the most important tasks to perform in unsupervised learning.

1.5 Reinforcement learning

In reinforcement learning, an agent changes its states to maximize its goals. There are four distinct concepts here: agent, state, action, and reward. Let's take a look at these in more detail:

1. Agent: This is the program we train. It chooses actions over time from its action space within the environment for a specified task.
2. State: This is the observation that's received by the agent from its environment and represents the agent's current situation.
3. Action: This is a choice that's made by an agent from its action space. The action changes the state of the agent.
4. Reward: This is the resultant feedback regarding the agent's action and describes how the agent ought to behave.

Reinforcement learning involves an agent, an environment, a set of actions, a set of states, and a reward system. The agent interacts with the environment and modifies its state. Based on this modification, it gets rewards or penalties for its input. The goal of the agent is to maximize the reward over time.

1.6 Difference between supervised and reinforcement learning

A supervised learning algorithm is used when we have a labeled training dataset. Reinforcement learning is used in a scenario where an agent interacts with an environment to observe a basic behavior and change its state to maximize its rewards or goal.

One of the essential features of RL is that the agent's action may not affect the immediate state of the environment that it is working in, but it can affect the subsequent states. Hence, the algorithm might not learn in the initial state but can learn after a few states are changed.

1.7 Applications of reinforcement learning

1. Text mining: Several researchers and companies have started to use the RL- based text generation model to produce highly readable text summaries from long text.

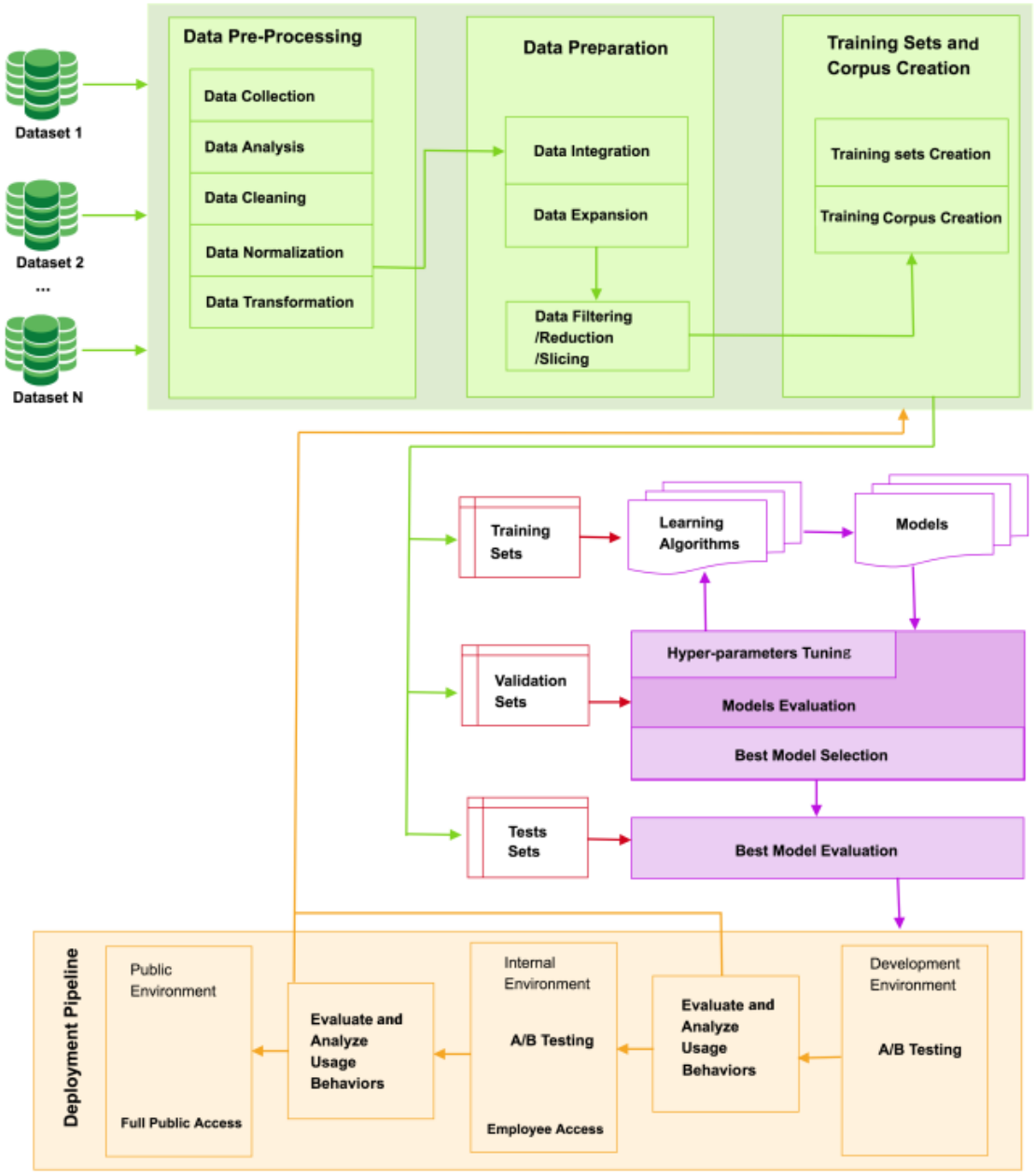
2. Robotics: Several deep RL-based algorithms are used in the field of robotics engineering to enhance the performance of robotics based on the reward system.
3. Healthcare: Several studies show that RL can be used in healthcare to optimize medication dosage and treatment policies.
4. Trading: Several RL-based models are used in trading businesses to optimize trading outcomes.

1.8 Unified machine learning workflow

The choice of what machine learning algorithm to use always depends on the type of data you have. If you have a labeled dataset, then your obvious choice will be to select one of the supervised machine learning techniques. Moreover, if your labeled dataset contains real values in the target variable, then you will opt for regression algorithms. Finally, if your labeled dataset contains a categorical variable in the target variable, then you will opt for the classification algorithm. In any case, the algorithm you choose always depends on the type of dataset you have.

The machine learning workflow can be divided into several stages:

1. Data preprocessing
2. Data preparation
3. Training sets and corpus creation
4. Model creation and training
5. Model evaluation
6. Best model selection and evaluation
7. Model deployment



1.8.1 Data preprocessing

Data preprocessing involves several steps, including data collection, data analysis, data cleaning, data normalization, and data transformation. The first step in data preprocessing is data collection.

1.8.2 Data collection

In data science, the most important thing is data. The data holds the ground truth about any events, phenomena, or experiments that are going on around us. Once we've processed the data, we get information. Once we've processed this information, we can derive knowledge from it. Hence, the most prominent stage in knowledge extraction is how relevant the data that's being captured is. There are different types of data, including structured data, unstructured data, and semi-structured

data. Structured data maintains a uniform structure in all the observations, similar to relational database tables. Unstructured data does not maintain any particular structure. Semi-structured data maintains some structure in the observation. JavaScript Object Notation (JSON) is one of the most popular ways to store semi-structured data. The process of collecting data in any company depends on the kind of project and the type of information that needs to be studied. The different types of datasets range from text data, file, database, sensors data, and many other Internet of Things (IoT) data.

1.8.3 Data analysis

This is one of the preliminary analysis phases where we perform exploratory data analysis to understand the dataset. This step tells us about the type of data we have at hand, the target variable, how many rows and columns we have in the data, the data types of each column, how many missing rows we have, what the data distribution looks like, and so on.

1.8.4 Data cleaning, normalization, and transformation

We discussed data cleaning, normalization, and data transformation in detail in Chapter 4, Data Transformation. We discussed how we can rescale the dataset, how we can convert the dataset into a standard dataset, how we can binarize data, and how we can perform one-hot encoding and label encoding. After all these three steps, our missing data will have been taken care of in terms of noisy data being filtered and inconsistent data being removed.

1.8.5 Data preparation

Sometimes, the dataset we have is not always in the right shape for it to be consumed by machine learning algorithms. In such conditions, data preparation is one of the most essential things we can do. We need to integrate data from several sources, perform slicing and grouping, and aggregate them into the correct format and structure. This step is referred to as data preparation.

1.8.6 Training sets and corpus creation

After the data preparation step, the resulting dataset is used as a training corpus. Generally, the training corpus is split into three chunks: a training set, a validation set, and a testing set. The training set is the chunk of data that you use to train one or more machine learning algorithms. The validation set is the chunk of data that you use to validate the trained model. Finally, the testing set is the chunk of data that you use to assess the performance of a fully trained classifier.

1.8.7 Model creation and training

Once we have split the dataset into three chunks, we can start the training process. We use the training set to construct the machine learning model. Then, we use the validation set to validate the model. Once the model has been trained, we use the test set to find the final performance of the model.

1.8.8 Model evaluation

Based on the performance of the test data, we can create a confusion matrix. This matrix contains four different parameters: true positive, true negative, false positive, and false negative. Consider the following confusion matrix:

- **True positives**: The model predicts TRUE when the actual value is TRUE.
- **True negatives**: The model predicts FALSE when the actual value is FALSE.
- **False-positives**: The model predicts TRUE when the actual value is FALSE. This is also referred to as a Type I Error.
- **False-negatives**: The model predicts FALSE when the actual value is TRUE. This is also referred to as a Type II Error.

This matrix shows four distinct parameters:

1. True positives: The model predicts TRUE when the actual value is TRUE.
2. True negatives: The model predicts FALSE when the actual value is FALSE.
3. False-positives: The model predicts TRUE when the actual value is FALSE. This is also referred to as a Type I Error.
4. False-negatives: The model predicts FALSE when the actual value is TRUE. This is also referred to as a Type II Error.

Once we know about the confusion matrix, we can compute several accuracies of the model, including precision, negative predicate value, sensitivity, specificity, and accuracy. Let's take a look at each of them, one by one, and learn how they can be computed.

The precision is the ratio of true positive and the sum of a true positive and false positive. The formula is as follows:

$$precision = \frac{TP}{(TP + FP)}$$

The formula for the **Negative Predictive Value** (**NPV**) is as follows:

$$NAV = \frac{TN}{(TN + FN)}$$

Similarity, the formula for **sensitivity** is as follows:

$$sensitivity = \frac{TP}{(TP + FN)}$$

The formula for specificity is as follows:

$$specificity = \frac{TN}{(TN + FP)}$$

Finally, the accuracy of the model is given by the formula as:

$$accuracy = \frac{TP + TN}{(TP + TN + FP + FN)}$$

Let's take a look at an example. Consider we built a supervised classification algorithm that looks at the picture of a window and classifies it as dirty or not dirty. The final confusion matrix is as follows:

	Predicted: Dirty	Predicted: Not dirty
Actual: Dirty	TP = 90	FN = 40
Actual: Not dirty	FP = 10	TN = 60

Now, let's compute the accuracy measures for this case:

Precision = TP / (TP + FP) = 90 /(90 + 10) = 90%. This means 90% of the pictures that were classified as dirty were actually dirty.

Sensitivity = TP / (TP + FN) = 90/(90 + 40) = 69.23%. This means 69.23% of the dirty windows were correctly classified and excluded from all non-dirty windows.

Specificity = TN / (TN + FP) = 60 / (10 + 60) = 85.71%. This means that 85.71% of the non-dirty windows were accurately classified and excluded from the dirty windows.

Accuracy = (TP + TN)/(TP + TN + FP + FN) = 75%. This means 75% of the samples were correctly classified.

Another commonly used accuracy model that you will encounter is the F1 Score. It is given by the following equation:

$$\text{F1 Score} = 2 \times \frac{precision \times recall}{precision + recall}$$

As we can see, the F1 score is a weighted average of the recall and precision. There are too many accuracy measures, right? This can be intimidating at the beginning, but you will get used to it over time.

1.8.9 Best model selection and evaluation

Model selection is an essential step in the machine learning algorithm workflow. However, model selection carries different meanings in different contexts:

1. Context 1: In the machine learning workflow context, model selection is the process of selecting the best machine learning algorithms, such as logistic regression, SVM, decision tree, Random Forest classifier, and so on.
2. Context 2: Similarly, the model selection phase also refers to the process of choosing between different hyperparameters for any selected machine learning algorithm.

In general, model selection is the method of choosing one best machine learning algorithm from a list of possible candidate algorithms for a given training dataset. There are different model selection techniques. In a normal scenario, we split the training corpus into a training set, a validation set, and a testing set. Then, we fit several candidate models on the training set, evaluate the models using the validation set, and report the performance of the model on the testing set. However, this scenario of model selection only works when we have a sufficiently large training corpus.

However, in many cases, the amount of data for training and testing is limited. In such a case, the model selection becomes difficult. In such a case, we can use two different techniques: probabilistic measure and resampling method. We suggest that you go through the Further reading section of this chapter if you wish to understand these model selection techniques.

1.8.10 Model deployment

Once you've got the best model based on your dataset and the model has been fully trained, it is time to deploy it. Showing how a model can be fully deployed into a working environment is beyond the scope of this book. You can find sufficient resources that will point you in the right direction in the Further reading section. The main idea regarding model deployment is to use the trained model in a real working environment. Once deployed, it should go through A/B user testing so that you know how it will work in a real scenario. Once it has been fully tested, the API can be made available to the public.

[ ]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import matplotlib.cm as cm
import seaborn as sns

sns.set()
plt.rcParams['figure.figsize'] = (14, 7)
```

[2]:
```
df = pd.read_excel("https://github.com/sureshHARDIYA/phd-resources/blob/master/
↪Data/Review%20Paper/acm/preprocessed.xlsx?raw=true")
df.head(10)
```

[2]:

	Unnamed: 0	…	Year
0	786	…	2016
1	885	…	2018
2	1083	…	2015
3	1004	…	2012
4	899	…	2013

```
5        1282  …   2016
6         434  …   2018
7        1168  …   2016
8        1238  …   2017
9        1272  …   2019

[10 rows x 7 columns]
```

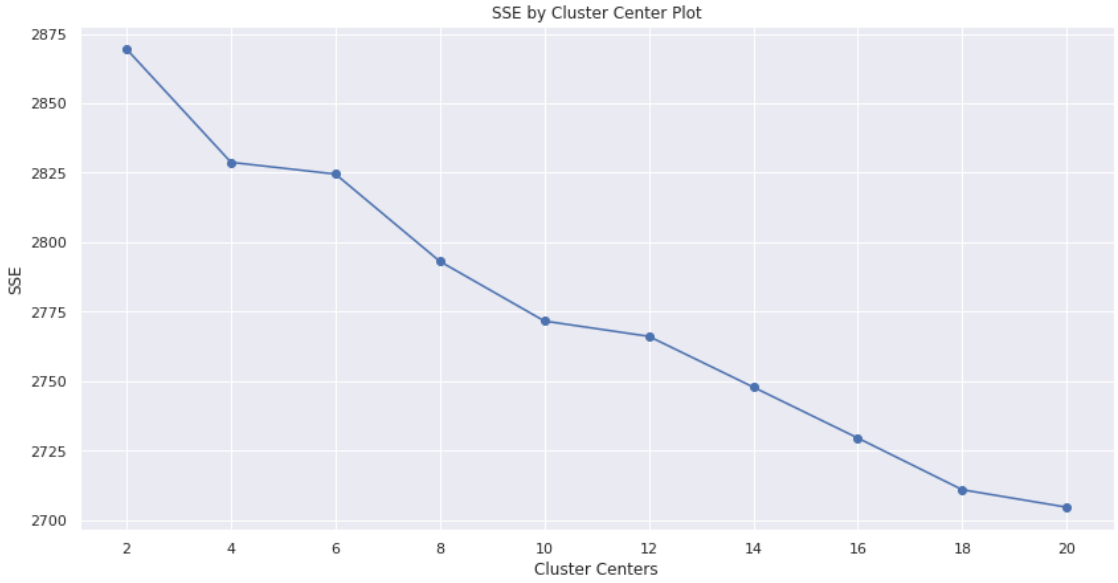
[ ]:
```
from sklearn.cluster import MiniBatchKMeans
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
```

[ ]:
```
tfidf = TfidfVectorizer(
    min_df = 5,
    max_df = 0.95,
    max_features = 8000,
    stop_words = 'english'
)
tfidf.fit(df.Title)
text = tfidf.transform(df.Title)
```

[5]:
```
def generate_optimal_clusters(data, max_k):
    iters = range(2, max_k+1, 2)

    sse = []
    for k in iters:
        sse.append(MiniBatchKMeans(n_clusters=k, init_size=1024,␣
 ↪batch_size=2048, random_state=20).fit(data).inertia_)
        print('Fitting {} clusters'.format(k))

    f, ax = plt.subplots(1, 1)
    ax.plot(iters, sse, marker='o')
    ax.set_xlabel('Cluster Centers')
    ax.set_xticks(iters)
    ax.set_xticklabels(iters)
    ax.set_ylabel('SSE')
    ax.set_title('SSE by Cluster Center Plot')

generate_optimal_clusters(text, 20)
```

```
Fitting 2 clusters
Fitting 4 clusters
Fitting 6 clusters
Fitting 8 clusters
Fitting 10 clusters
Fitting 12 clusters
```

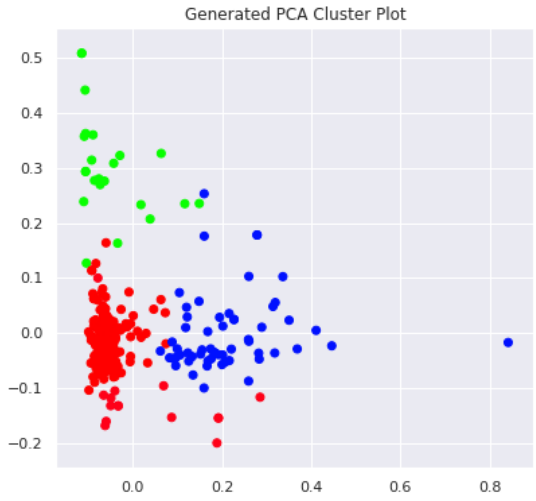
```
Fitting 14 clusters
Fitting 16 clusters
Fitting 18 clusters
Fitting 20 clusters
```

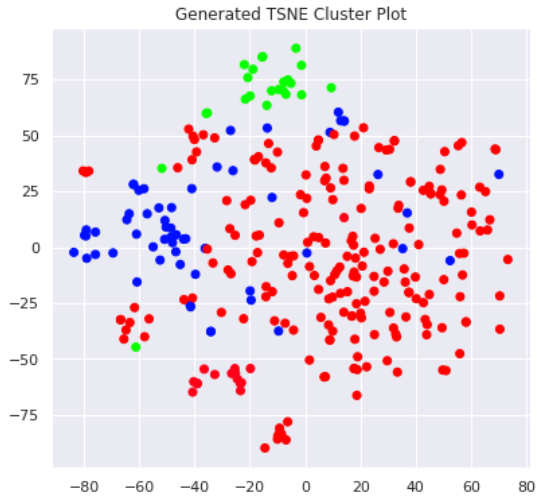


[ ]:
```
clusters = MiniBatchKMeans(n_clusters=4, init_size=1024, batch_size=2048,␣
↪random_state=20).fit_predict(text)
```

[8]:
```
max_label = max(clusters)
max_items = np.random.choice(range(text.shape[0]), size=3000, replace=True)
pca = PCA(n_components=2).fit_transform(text[max_items,:].todense())
tsne = TSNE().fit_transform(PCA(n_components=50).fit_transform(text[max_items,:
↪].todense()))

idx = np.random.choice(range(pca.shape[0]), size=300, replace=True)
label_subset = clusters[max_items]
label_subset = [cm.hsv(i/max_label) for i in label_subset[idx]]

f, ax = plt.subplots(1, 2, figsize=(14, 6))
ax[0].scatter(pca[idx, 0], pca[idx, 1], c=label_subset)
ax[0].set_title('Generated PCA Cluster Plot')

ax[1].scatter(tsne[idx, 0], tsne[idx, 1], c=label_subset)
ax[1].set_title('Generated TSNE Cluster Plot')
```

[8]:
```
Text(0.5, 1.0, 'Generated TSNE Cluster Plot')
```





[11]:
```
from wordcloud import WordCloud

fig, ax = plt.subplots(4, sharex=True, figsize=(15,10*4))

plt.rcParams["axes.grid"] = False

def high_frequency_keywords(data, clusters, labels, n_terms):
    df = pd.DataFrame(data.todense()).groupby(clusters).mean()

    for i,r in df.iterrows():
      words = ','.join([labels[t] for t in np.argsort(r)[-n_terms:]])
      print('Cluster {} \n'.format(i))
      print(words)
      wordcloud = WordCloud(max_font_size=40, collocations=False, colormap =␣
 ↪'Reds', background_color = 'white').generate(words)
      ax[i].imshow(wordcloud, interpolation='bilinear')
      ax[i].set_title('Cluster {} '.format(i), fontsize = 20)
      ax[i].axis('off')
high_frequency_keywords(text, clusters, tfidf.get_feature_names(), 50)
```

```
Cluster 0

bipolar,patient,framework,evaluation,risk,older,internet,healthcare,activity,app
roach,online,anxiety,research,digital,children,assessment,clinical,dementia,adap
tive,cognitive,intervention,disorders,technology,learning,psychiatric,community,
interventions,management,therapy,review,adults,use,support,designing,schizophren
ia,stress,data,people,analysis,care,self,mobile,disorder,using,patients,design,s
tudy,treatment,based,depression
Cluster 1
```

cessation,brief,comparing,single,disorder,people,adults,symptoms,risk,clinical,w
omen,prevention,reduce,improve,training,use,results,online,personalized,internet
,cluster,alcohol,anxiety,feedback,efficacy,patients,health,mental,therapy,primar
y,help,self,program,care,effects,cognitive,pilot,treatment,depression,tailored,e
ffectiveness,web,based,randomised,study,intervention,protocol,randomized,control
led,trial
Cluster 2

qualitative,physical,digital,implementation,self,medical,management,patient,adul
ts,designing,life,quality,work,development,systems,data,related,children,persons
,support,online,analysis,assessment,information,intervention,veterans,service,de
sign,patients,problems,behavioral,using,research,systematic,disorders,use,interv
entions,primary,treatment,based,study,services,review,severe,people,community,il
lness,care,mental,health
Cluster 3

modeling,implications,ethical,emotion,behavioral,dementia,based,young,designing,
homeless,dynamics,group,experiences,robot,predicting,mobile,game,depression,unde
rstanding,physical,people,challenges,therapy,study,patients,management,technolog
y,impact,technologies,self,anxiety,use,skills,interaction,networking,personal,di
sclosure,sites,data,networks,disclosures,using,design,online,network,support,men
tal,health,media,social

Cluster 0

Cluster 1

Cluster 2

Cluster 3